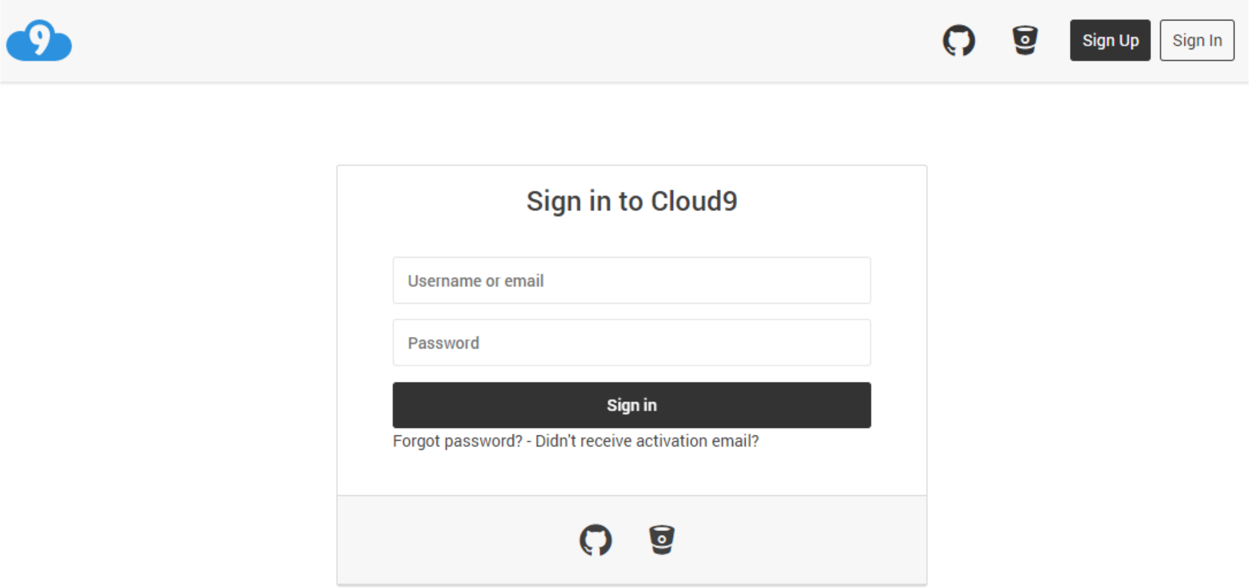
CSE575-Section2: Announcement Activity

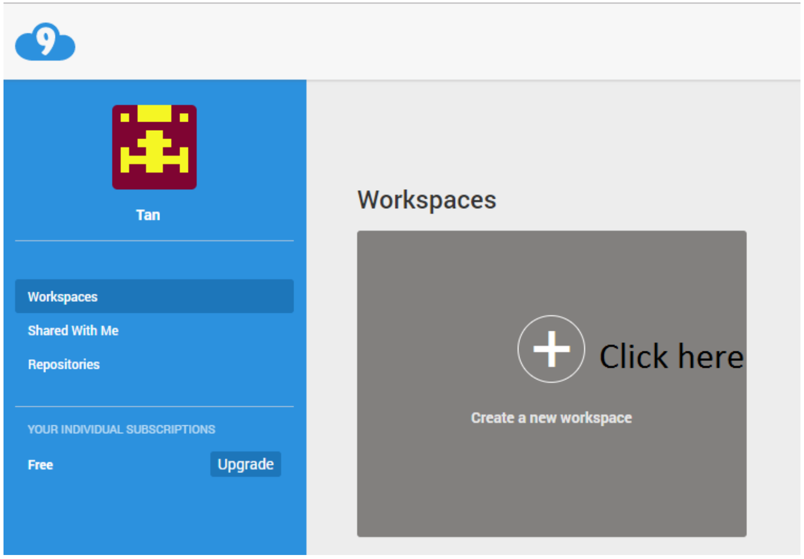
I. This activity will require a backend server to update the announcement online without having to update the app.

1. To create a backend server, I will use https://c9.io/ to host the server. This one is a free hosting site. We just need create an account.
2. Follow these steps:

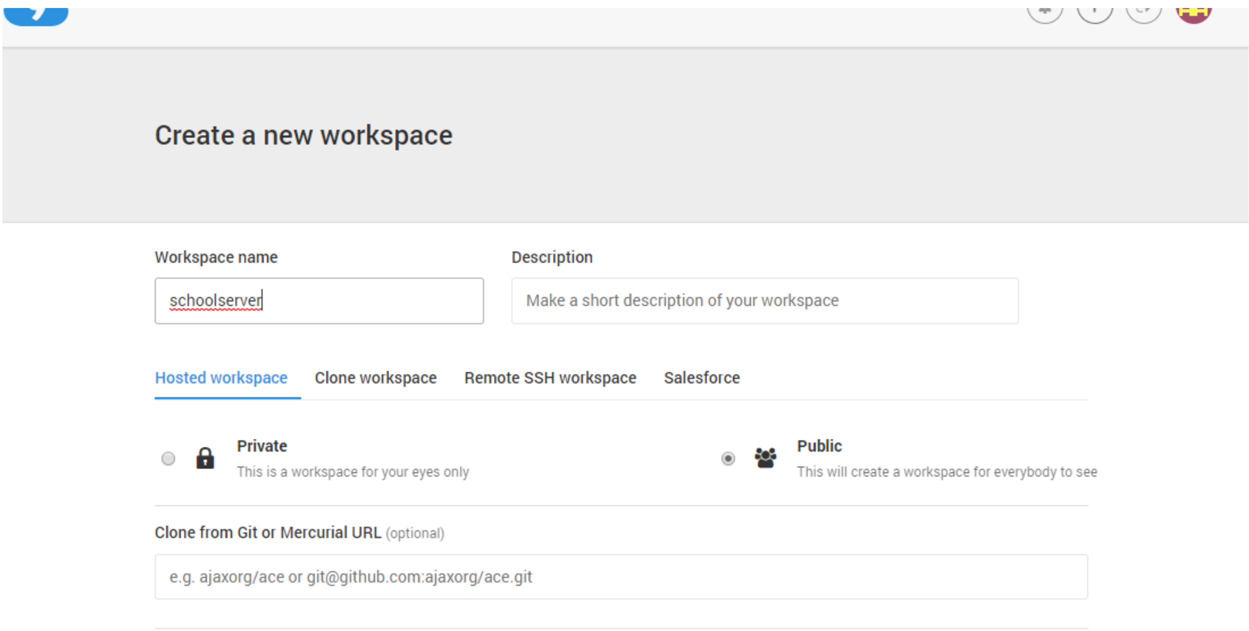
- Sign in after finishing the sign up process



- Click the create new workspace



- Name the workspace something that meaningful



Create a new workspace

Workspace name

schoolserver

Description

Make a short description of your workspace

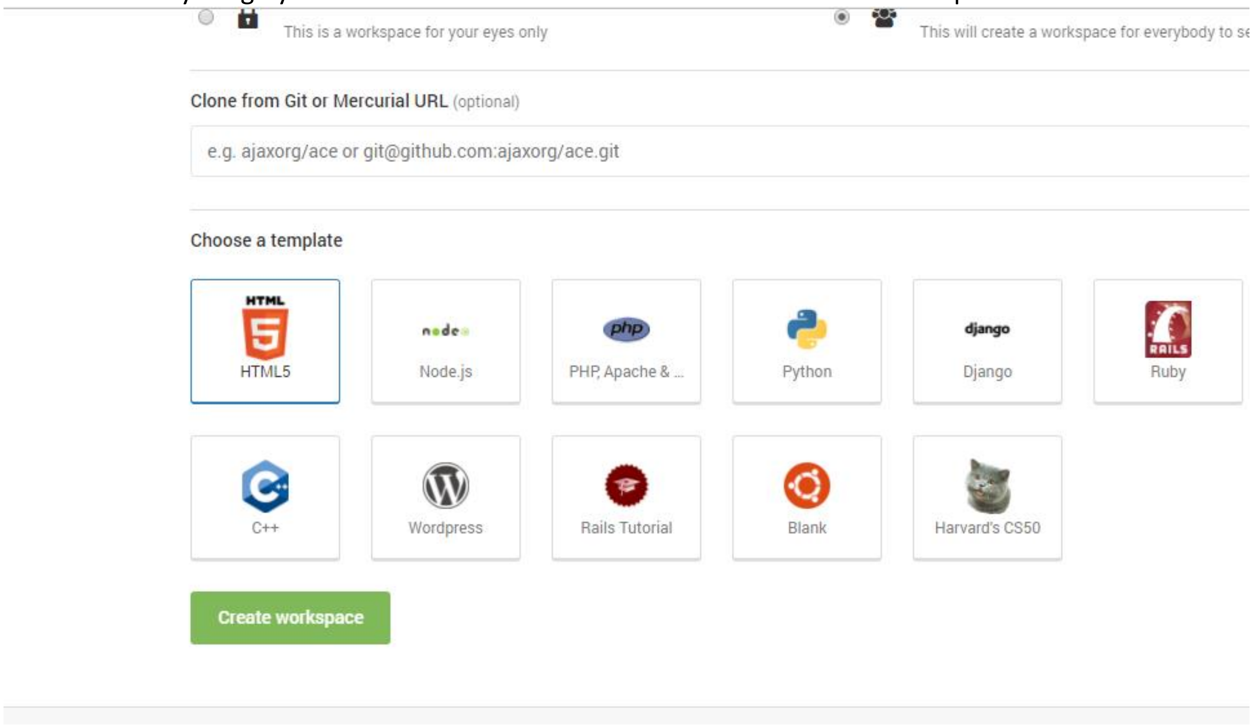
○ Private
This is a workspace for your eyes only

● Public
This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional)

e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

- Leave everything by default > scrolldown to the bottom > click create workspace



○ This is a workspace for your eyes only

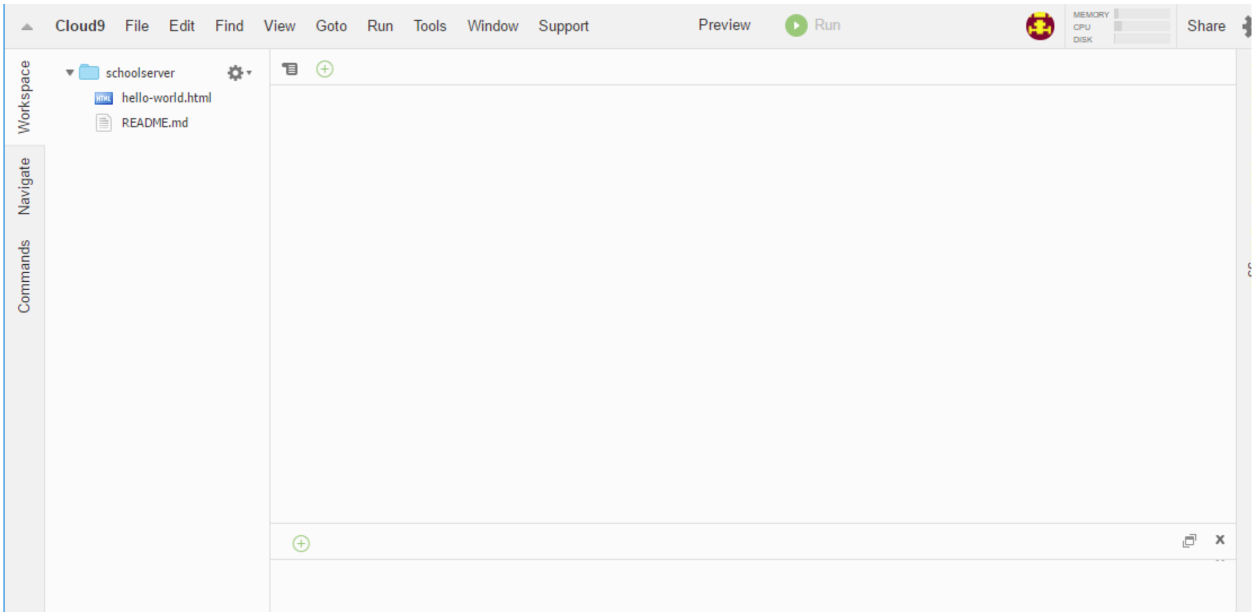
● This will create a workspace for everybody to se

Clone from Git or Mercurial URL (optional)

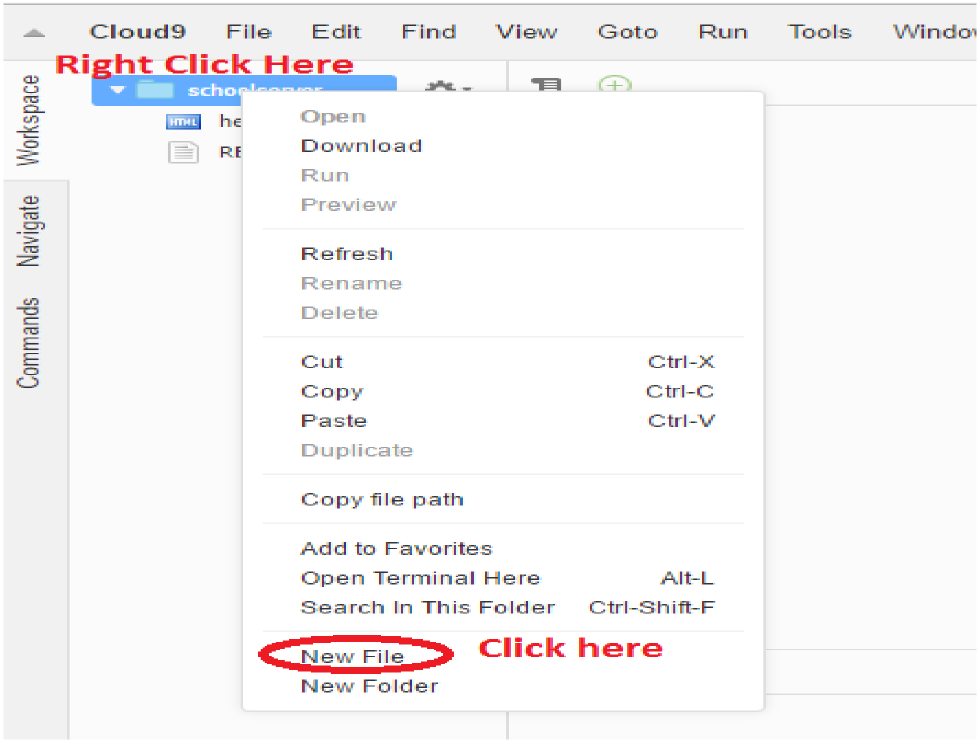
e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Choose a template

- Your workspace will look like this



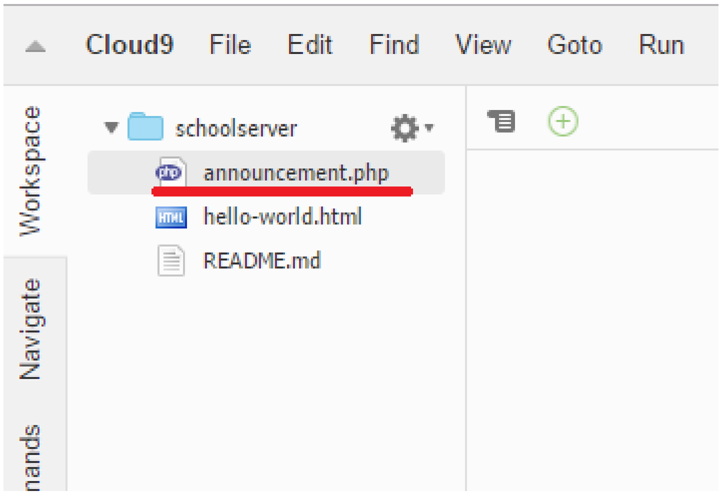
3. Now, we will create PHP file to output the JSON strings that we will use for announcement activity



Right Click Here

Click here

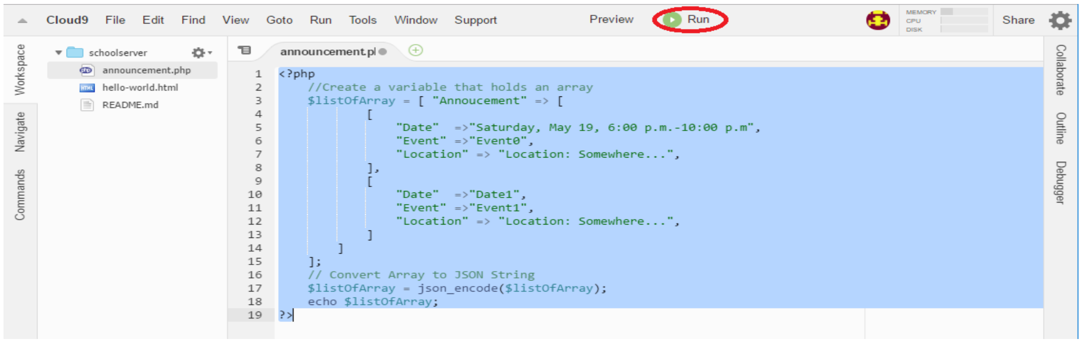
I named the new file as announcement.php



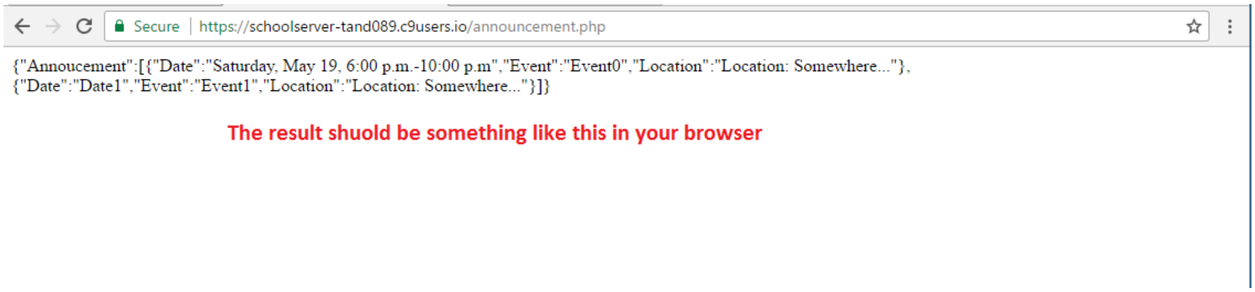
Add following line of codes into the announcement.php file

```
<?php
   //Create a variable that holds an array
   $listOfArray = [ "Annoucement" => [
         [
            "Date"  =>"Saturday, May 19, 6:00 p.m.-10:00 p.m",
            "Event" =>"Event0",
            "Location" => "Location: Somewhere...",
         ],
         [
            "Date"  =>"Date1",
            "Event" =>"Event1",
            "Location" => "Location: Somewhere...",
         ]
      ]
   ];
   // Convert Array to JSON String
   $listOfArray = json_encode($listOfArray);
   echo $listOfArray;
?>
```

Click Run button to see the result



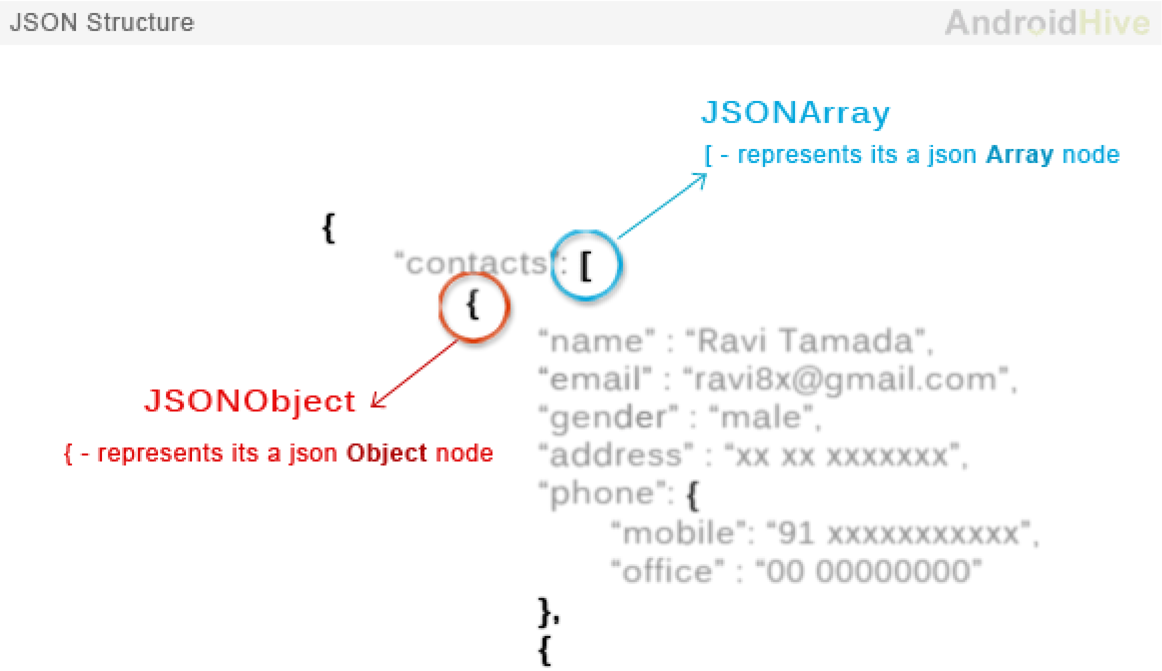
```
<?php
    //Create a variable that holds an array
    $listOfArray = [ "Annoucement" => [
            [
                "Date"  =>"Saturday, May 19, 6:00 p.m.-10:00 p.m",
                "Event" =>"Event0",
                "Location" => "Location: Somewhere...",
            ],
            [
                "Date"  =>"Date1",
                "Event" =>"Event1",
                "Location" => "Location: Somewhere...",
            ]
        ]
    ];
    // Convert Array to JSON String
    $listOfArray = json_encode($listOfArray);
    echo $listOfArray;
?>
```



```
{"Annoucement":[{"Date":"Saturday, May 19, 6:00 p.m.-10:00 p.m","Event":"Event0","Location":"Location: Somewhere..."},
{"Date":"Date1","Event":"Event1","Location":"Location: Somewhere..."}]}
```

The result shuold be something like this in your browser

*Note: here is an json structure, that we will use as a reference for future use.


JSON Structure
AndroidHive
JSONArray
[- represents its a json Array node
JSONObject
{ - represents its a json Object node

```
{
    "contacts": [
        {
            "name" : "Ravi Tamada",
            "email" : "ravi8x@gmail.com",
            "gender" : "male",
            "address" : "xx xx xxxxxxx",
            "phone": {
                "mobile": "91 xxxxxxxxxx",
                "office" : "00 00000000"
        },
        {
```

II. After finishing the backend server, we will build the frontend in our app to retrieve the json strings from server. I will use AsyncTask method in this case.

1. HTTP AsyncTask is a commonly needed function in most Android app. So, it is better to have a class to give us a reusable codes that can take care of the AsyncTask process in when we need it.

    i. Create a class called HttpHandler.java

<u>HttpHandler.java: </u>details explain in the comments

```
package com.tando.school;

import android.util.Log;

import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.ProtocolException;
import java.net.URL;
```

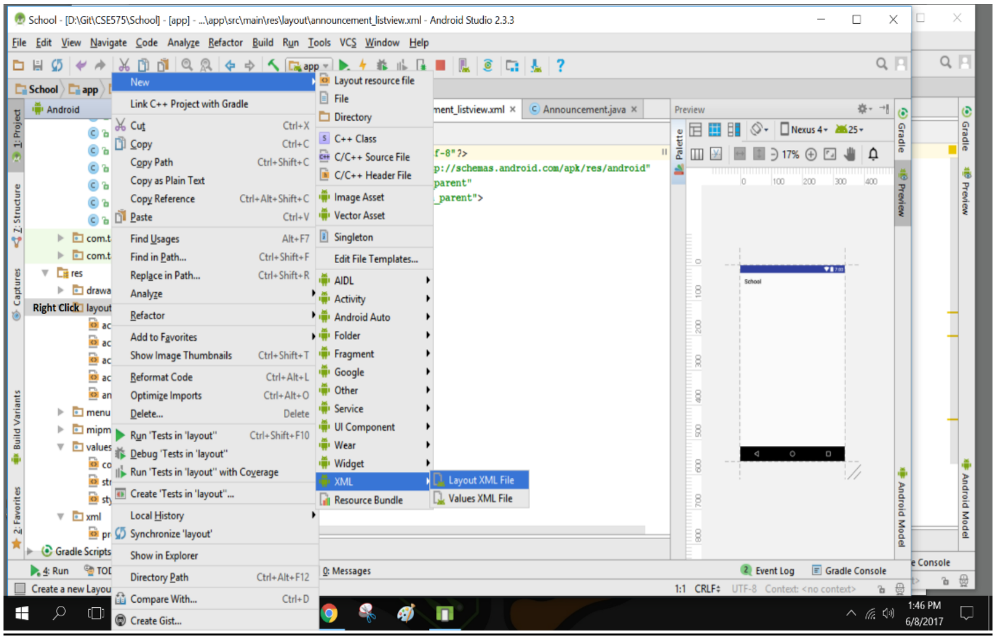
```java
/**
 * Created by tan089 on 6/8/2017.
 */

public class HttpHandler {
    //returns the simple name of the underlying class, easier to track in the
Android monitor
    private static final String TAG = HttpHandler.class.getSimpleName();
    public HttpHandler() {
    }
    public String makeServiceCall(String reqURL) {
        String response = null;
        try {
            //Creates a URL from the given String
            URL url = new URL(reqURL);
            //open the connection to the url object
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            //use GET method to get the JSON from server
            conn.setRequestMethod("GET");
            // read the response from server
            InputStream in = new BufferedInputStream(conn.getInputStream());
            response = convertStreamToString(in);
            //some Exceptions
        } catch (MalformedURLException e) {
            Log.e(TAG, "MalformedURLException: " + e.getMessage());
        } catch (ProtocolException e) {
            Log.e(TAG, "ProtocolException: " + e.getMessage());
        } catch (IOException e) {
            Log.e(TAG, "IOException: " + e.getMessage());
        } catch (Exception e) {
            Log.e(TAG, "Exception: " + e.getMessage());
        }
        return response;
    }
    private String convertStreamToString(InputStream is) {
        //BufferedReader reads text from a character-input stream
        BufferedReader reader = new BufferedReader(new InputStreamReader(is));
        //This constructs a string builder with no characters in it
        StringBuilder sb = new StringBuilder();

        String line;
        try {
            //To convert the InputStream to String we use the
BufferedReader.readLine()
            while ((line = reader.readLine()) != null) {
                //Each line will appended to a StringBuilder and returned as
String.
                sb.append(line).append('\n');
            }
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            //close when no more input data stream
            try {
                is.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        //return string
        return sb.toString();
    }
}
```

2. Create a new empty activity called Announcement. Using the same method as I have discussed on the last section.
   Note*: remember to go back to the AndroidManifest.xml to create label, add theme, and make a back button.

- Style the UI for announcement activity.
  Activity_annoucement.xml

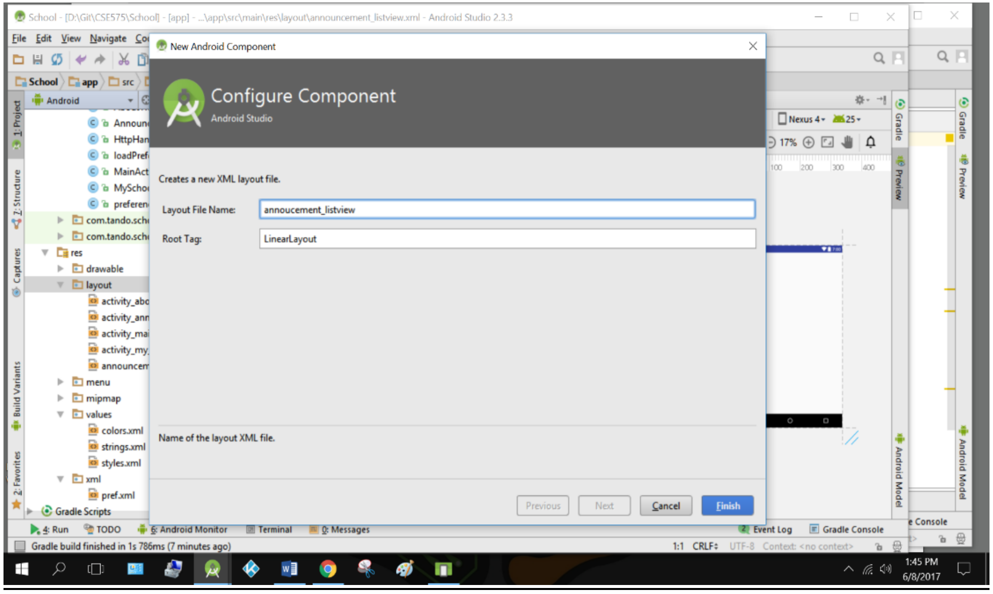
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.tando.school.Announcement"
    android:orientation="vertical"
    android:background="#FFF">
    <!--Logo -->
    <ImageView
        android:id="@+id/logo"
        android:layout_width="150dp"
        android:layout_height="100dp"
        android:background="@drawable/school_logo"
        android:layout_gravity="center"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Announcement"
        android:textColor="#000"
        android:textSize="30dp"
        android:textStyle="bold"
        android:gravity="center"/>
    <ListView
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:id="@+id/announcList">
    </ListView>
</LinearLayout>
```

- Create new xml layout to style the listview. I named it announcement_listview



School - [D:\Git\CSE575\School] - [app] - ...\app\src\main\res\layout\announcement_listview.xml - Android Studio 2.3.3

```
f-8"?>
p://schemas.android.com/apk/res/android"
parent"
_parent">
```



School - [D:\Git\CSE575\School] - [app] - ...\app\src\main\res\layout\announcement_listview.xml - Android Studio 2.3.3

New Android Component

Configure Component

Android Studio

Creates a new XML layout file.

Layout File Name: annoucement_listview

Root Tag: LinearLayout

Name of the layout XML file.

Gradle build finished in 1s 786ms (7 minutes ago)

Announcement_listview.xml

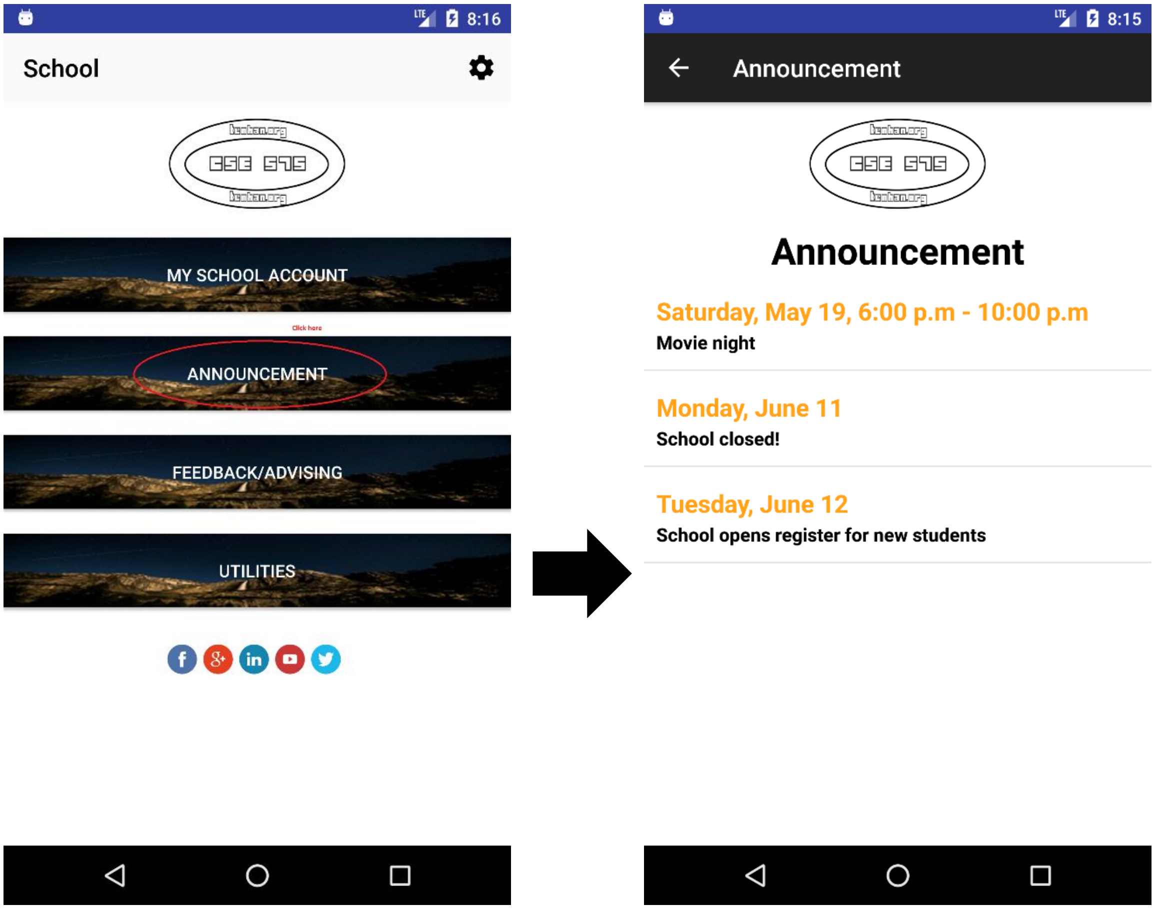
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="10dp">
    <TextView
        android:id="@+id/aDate"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="2dp"
        android:paddingTop="6dp"
        android:textColor="#ffa31a"
        android:textSize="20dp"
        android:textStyle="bold"
        android:text="Date: "/>
    <TextView
        android:id="@+id/aEvent"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="2dip"
        android:textColor="#000"
        android:textSize="15dp"
        android:text="Note: "
        android:textStyle="bold" />
</LinearLayout>
```

- Next, go to Announcement.java to make the activity functioning. Details of the codes are explained in the comments.

Announcement.java

```
package com.tando.school;

import android.app.ProgressDialog;
import android.os.AsyncTask;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.Toast;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;
import java.util.HashMap;

public class Announcement extends AppCompatActivity {
    //returns the simple name of the underlying class, easier to track in the Android
monitor
    private String TAG = Announcement.class.getSimpleName();
    //Prpgress bar while retrieving data
    private ProgressDialog pDialog;
    //Declare ListView
    private ListView AnnouncementList;
    // URL to get calendar JSON
    private static String url ="https://schoolserver-
tand089.c9users.io/announcement.php";
    //Declare an array to store the list of items
    ArrayList<HashMap<String, String>> announcList;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_announcement);

        announcList = new ArrayList<>();

        AnnouncementList = (ListView) findViewById(R.id.announcList);

        //execute the GetEvents class
        new GetEvents().execute();
    }
    /**
     * Async task class to get json by making HTTP call
     */
    //Create a GetEvents class to make http calls on background thread
    private class GetEvents extends AsyncTask<Void, Void, Void> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            // Showing progress dialog
            pDialog = new ProgressDialog(Announcement.this);
            pDialog.setMessage("Loading...");
            pDialog.setCancelable(false);
            pDialog.show();
        }

        @Override
        protected Void doInBackground(Void... arg0) {
            //call the HttpHandler class
            HttpHandler httpHandler = new HttpHandler();
            // Making a request to url and getting response
            String jsonStr = httpHandler.makeServiceCall(url);
            //make a log to check for response or error
            Log.e(TAG, "Response from url: " + jsonStr);
            //Get JSON
            if (jsonStr != null) {
                JSONObject jsonObj = null;
                try {
                    jsonObj = new JSONObject(jsonStr);
                    /* Getting JSON Array node
                    * Note: the Annoucement is the object node in our JSON. Check the
JSON */
                    JSONArray calendarList = jsonObj.getJSONArray("Annoucement");

                    // looping through All Events
                    for (int i = 0; i < calendarList.length(); i++) {
                        JSONObject c = calendarList.getJSONObject(i);
                        //get string from the json file
                        String Date = c.getString("Date");
                        String Event = c.getString("Event");


                        // tmp hash map for single event
                        HashMap<String, String> announcements = new HashMap<>();

                        // adding each child node to HashMap key => value
                        announcements.put("Date", Date);
                        announcements.put("Event", Event);


                        // adding contact to announcList
                        announcList.add(announcements);
                    }
                } catch (final JSONException e) {
                    Log.e(TAG, "Json parsing error: " + e.getMessage());
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            Toast.makeText(getApplicationContext(),
                                    "Json parsing error: " + e.getMessage(),
                                    Toast.LENGTH_LONG)
                                    .show();
                        }
                    });
                }
            } //End If
            else {
                Log.e(TAG, "Couldn't get json objects from server.");
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(getApplicationContext(),
                                "errors!",
                                Toast.LENGTH_LONG)
                                .show();
                    }
                });

            }
            return null;
        } //End doing background

        @Override
        protected void onPostExecute(Void result) {
            super.onPostExecute(result);
            // Dismiss the progress dialog
            if (pDialog.isShowing()) {
                pDialog.dismiss();
            }
            /**
             * Updating parsed JSON data into ListView
             * */
            ListAdapter adapter = new SimpleAdapter (
                Announcement.this, announcList, R.layout.announcement_listview, new
String[]{"Date", "Event"}, new int[]{R.id.aDate,
                    R.id.aEvent});
            AnnouncementList.setAdapter(adapter);

        }
    } //End AsyncTask
}
```

3. Hit the run app button to see the result



Announcement

Saturday, May 19, 6:00 p.m - 10:00 p.m

Movie night

Monday, June 11

School closed!

Tuesday, June 12

School opens register for new students